User Profile Based Activities in Flexible Processes

Marco Fisichella, Juri Luca de Coi, Ricardo Kawase, Maristella Matera

User Profile Based Activities in Flexible Processes

Marco Fisichella
L3S Research Center
Hannover, Germany
fisichella@L3S.de

Ricardo Kawase
L3S Research Center
Hannover, Germany
kawase@L3S.de

Juri Luca De Coi
University Jean Monnet
Saint-Etienne, France
juri.luca.decoi@univ-st-etienne.fr

Maristella Matera
Politecnico di Milano, DEI
Milan, Italy
matera@elet.polimi.it

ABSTRACT

COOPER platform is a collaborative, open environment that leverages on the idea of flexible, user-centric process support. It allows cooperating team members to define collaborative processes and flexibly modify the process activities even during process execution. In this paper we describe how the incorporation of decentralized user data through mashups, allows the COOPER platform to support the definition and execution of the so called user profile based activities, i.e., process activities that are adapted based on the preferences of the process actors. We define two basic types of user profile based activities, namely user adapted activities and user conditional activities. The first are modeled according to the user profile data, while the second employs the same user data to enable automatic workflow decisions.

Categories and Subject Descriptors

D.2.2 [**Software Engineering**]: [Design Tools and Techniques, Modules and interfaces, Software libraries]; H.5.2 [**Information Interfaces and Presentation**]: [User Interfaces, Graphical user interfaces, Interaction styles]; H.5.4 [**Information Interfaces and Presentatio**]: [Hypertext Hypermedia, Architectures]

Keywords

Flexible Processes, Mashups, Web Application Design, Atomic Activities, User profiles, Adaptive Activities.

1. INTRODUCTION

Workflow Management Systems (WfMSs) support the definition and the execution of business processes. Workflow applications are typically well-defined and highly repetitive in nature. They are based on process models, defined at design time by expert designers to capture constraints on the execution of tasks, their assignment to end-users and their mapping to resources.

Based on such pre-defined models, WfMSs then manage task enactment at runtime, by instantiating several concurrent processes (the so called cases), each one characterized by its own set of input parameters, end-users, and resources.

While the explicit definition of business processes is becoming more and more popular in the early phases of software engineering, WfMSs cannot claim a comparable success and popularity. Key problems are the intrinsic complexity of the WfMSs themselves, but especially the excess of rigidity of workflow enactment, which leaves no freedom to end-users to 'personalize' processes and process activities to their specific needs, which enlarges the gap between business processes models, humans and their activities [3]. This is especially true in all those organizational contexts where collaboration takes place within teams of people that have to coordinate to reach a given goal, e.g., the release of some artifact. One can think for example to the development of team-based projects within a company, but also to the so-called 'learning-by-doing', an increasingly diffused form of learning through which individuals (e.g., students in university, employees in organizations in different domains, company partners and customers, etc.) learn by working on a project and sharing their activities with others peers.

In all the previous cases, collaboration processes are difficult to predict completely in advance, and should be flexible enough to be adapted to the preferences of the individual actors, and to the evolution of background knowledge and competences. This is especially true when coordination of activities from different actors is also required and implies the definition of processes [3, 21]. Such processes are 'lightweight', meaning that they do not show an intrinsic complexity and, since they can hardly be completely predefined at design time, generally exhibit an explosive number of alternatives that depend on the specific needs of the involved actors. Such alternatives may refer both to the coordination flow of the different activities as well as to single activities,

WIMS'12, June 13-15, 2012 Craiova, Romania.

which actors might want to configure and 'personalize' by themselves. Such processes therefore escape the ability of being fully modeled, as it happens for traditional business processes [7].

At a larger vision, flexibility may also be intended as the capability of a system to automatically discovery (and integrate) new services/processes via intelligent methodologies (e.g., [13, 14]).

To provide personalized activities means to understand the user. User profiling has evolved to a mature technology that is presented to us on a daily basis, especially in the Web as we navigate and submit our personal information and opinions to several search engines, e-commerce systems and social networking sites that build the users profile. In this field, collaborative filtering and social network analysis [17] has been identified as the most prominent techniques on user profiling. Fortunately most of these services provide APIs for accessing the user information that they hold on their servers (once approved by the user), giving to supplementary systems the possibility to reuse this data. The smart use of this data provides at the same time broader and more refined user profiles. In COOPER, we propose an approach that facilitates: i) at design time, the definition of the activity, and ii) at execution time, the user-based customization of such adaptive activities. Especially considering that software (and Web) development is increasingly moving toward the possibility offered to users to mash-up their applications [4], our approach enables the composition of activity through the reuse of ready to use resources (mashup components), available in a local repository or even scouted on the Web, creating a cross-application environment and enabling the data interoperability from different resources.

The same mash-up paradigm is also offered to the end users to customize their activities at execution time. The proposed mechanism for the mash-up of activity capitalizes on our previous experience on the development of mashup environments, and its feasibility is therefore proved through the adoption of the Mixup tool for mashup creation [24, 16].

Creating Mashups is the new trend of combining multiple Web 2.0 applications where the ability to share user profiles becomes essential for a better integration and cooperation between single applications. By using mashed-up activities COOPER achieves different levels of benefits: first during design time by enabling the reuse of existing services and cross application data; second, during running time, by supporting user profiling and adaptive activities.

In this paper, we capitalize on a reference model for flexible processes definition and execution, COOPER [11], which effectively supports the dynamic, user-based management of adaptive collaboration processes. The main merit of the COOPER approach is that users (i.e., process actors) are provided with the ability to self-organize adaptive processes that can be executed on the Web, by using a simple Web interface, process templates, and an extensible library of activity types (mashups) already equipped with their Web front-end, that can be used to personalize process templates and create or modify processes even at run-time.

2. RELATED WORKS

The most notable industrial efforts so far devoted to enhance process flexibility have been done in the field of groupware. Such applications support individual tasks (especially based on email facilities), but offer very limited support to sustain collaborative processes. They indeed provide a set of hard-wired collaboration activities, and the extension of this set to respond to unanticipated collaboration needs is difficult or even impossible. This scenario often boosts end-users to migrate from activities coded inside processes to short-lived stand alone applications [12], disconnected from the processes and therefore completely out of control without any possibility to carry and share the information the user creates while working. This reality therefore also complicates the will of team members to harvest knowledge and experience captured by their peers within their organization in any available format.

Very few research approaches have so far dealt with the management of dynamic and flexible processes, focusing on the ability of the process to be (partially) sketched at design time, but not completely specified until runtime, and modified during its execution [7, 1, 8, 10].

Other related workflow management systems develop on the idea of flexible and adaptive workflows [18, 20]. The basic approach is to design rule-based workflows that consider the outcome of the previous activities to guide the process in different branches of the workflow. It is aligned with the ideas in COOPER, however this adaptation is on the workflow level and not in the activity level. In these systems the activities are not adaptive themselves. Differently from COPPER, the possible workflow activities must be pre defined on design time. In addition they fall short to connect with the existing data available in the Web, thus lacking potential and beneficial interoperability.

As mentioned, though many systems like COOPER support flexible and extensible workflows, none has integrated the user modeling techniques provided by nowadays Web 2.0 applications to explicit influence process activities and process decisions. Beyond the benefits of flexible dynamic processes we aim to provide adaptive processes and activities through the employment of existing available user data, exploiting online existing Web 2.0 resources.

Whereas most of the Web services provide access to the user data and the exchange of login credentials is already facilitated by initiatives such as OpenID , still in most cases, users need to build their user profiles from scratch for every application. The combination of the different user profiles can be easily managed by the so-called mashup applications.

To facilitate the exchange and the interoperability of user profiles, a common semantic is needed to align the different data descriptions [4]. General User Model Ontology (GUMO) [19] or Friend of a Friend (FOAF) [9] are possible options for this task, as both descriptions aim to generalize and unify user profiles. However, pre-defined and static user profile ontologies will never cover the diverse needs of user's customizable applications since ontologies have the natural growing behavior over time. As a result we believe that such shared models should not follow specific ontologies restrictions, instead it should be grown upon, starting from successful implementations in specific systems [22].

Up to date works have discussed the best possible approaches to integrate decentralized user profiles [2] involving, first, a lingua franca, an agreement between all parties on a common representation and semantics [23]. However due to the amount of systems and different user models such model has never been wide accepted as seen in CUMULATE [25] or PersonIs [6]. A second, more flexible, option consists in the conversion of the user profile data stream. This allows different information providers to communicate on the same platform wherein the result information is not restricted to one specific set of systems [5]. As we have seen in [2] we can achieve this goal by simply integrating mashup components.

3. THE COOPER PLATFORM

COOPER is a collaborative, open environment that leverages on the idea of flexible, user-centric process support. Its most salient feature is that it allows cooperating team members (i.e., the platform end-users) to dynamically define collaborative processes, on the basis of the team's preferred procedures, and easily modify the planned processes, to cope with the evolution of individuals as well as of the whole team.

Giving the end-users the possibility to define and modify their processes requires the system to offer easy-to-use definition interfaces, able to guide team members in the composition of processes without requiring any specific knowledge and expertise on process design. Guiding inexperienced users requires that the system be 'aware' of the semantics of the domain where processes must be executed. Such awareness can be achieved by means of libraries of pre-defined activity types, able to reflect and support the possible tasks that users might need to coordinate and execute in a given context. In this section we will further describe the architecture, the concepts behind COOPER platform and how to employ external data streams as for example the user profile data for user profile based activities.

3.1 Activities

All the activities provided in COOPER platform in our previous works were predefined atomic activities that aimed to cover all the possible tasks that are performed on a regular basis during project work and that could be reusable in several process contexts. Our atomic activities library could be classified in four different categories, namely teamwork planning, resource management, communication, and reviewing and assessing [10]. However, due to the uncountable possibilities of process definitions, a closed corpus of activities will never be enough to support all the users' requirements. In this sense, to make the model and the platform more flexible we extend the atomic activities idea into customable activities. Therefore users may define their own activities for their processes. It is up to the model to support such flexibility, and up to the platform to provide easy means to create them.

We define an activity as a tuple

$T =< Name, HT, Pi >$, where:

- Name is the name of the domain-specific activity type;
- HT is the activity type's hypertext front-end, which is used as user interface for the execution of the activity. Each time an instance of a particular activity type is executed, the user is provided with the predefined hypertext portion;
- Pi represents the set of properties that characterize the activity (e.g., start and end date).

Activity types therefore represent the definition of process tasks, that are regularly performed by users to collaborate, and that can be used for the definition of collaborative processes. Their definition implies associating the task with a hypertext front-end, which is used as user interface for the execution of the activity. Some atomic activities have a general nature (e.g., those related to the management of documents), and can therefore be adopted in several domains where collaborative processes are required. Some other activities may however be particular for specific contexts and their identification requires an investigation of the addressed domain.

Starting from a library of activity types, the system guides the composition of sound, 'well-structured' processes [15]. In [11] we show how the offered mechanisms for process definition guarantee the semantically correct execution and termination of process instances, and the possibility to easily (flexibly) modify processes even during runtime. Providing guarantees on the process semantics aims at assisting the continuous re-definition or evolution of running processes by users that in most cases are inexperienced.

3.2 User profile based activities

So far the concerns of the model have been on the user-activities modeling process and user's interactions. Combining the new flexibility of customizable tasks through mashups and the maturity of user modeling and Web personalization technologies, COOPER offers a transparent method of creating user profile based activities. Former activities in the

platform relied primarily in user's interactions, i.e. whether the user completed a task in the workflow process or not. With the new approach, we bring to the platform, activities that first adapt to the user, and second, conditional activities that may not require explicit user interactions. In both cases it depends only on the user's characteristics i.e. the user's profile.

Although it may be seen as an obvious possibility - to create tasks that vary according to each user - this feature was not transparent to the users during the designing process and lacked the interoperability that can be provided by the employment of flexible mash-ups of RSS-based user data streams. During design time the user can explicitly create activities that are based on the user's profile that will handle the task. User adapted activities consist in tasks that require the user profile data stream as input. This input will adapt a task according to user's preferences, characteristics, past activities or any profile data. Each task may be already preconfigured for each user with predefined inputs and/or pre defined courses of actions.

A second type of user profile based activities is the User conditional activities. These activities not necessarily require user interaction. They describe activities that restrict user access according to his/her profile (e.g. user profile determines that he/she may not have access to a user conditional task until some level of requirement is achieved). Furthermore these activities also define workflow branches to be followed by each user (again according to the user profile). Once the platform is a collaborative environment, many users interact with the same process workflow and it is indispensable that each one have a different set of personal tasks and activities.

Instead of building our own user profile which may requires a lot of input and repetitive work from the users, providing activities that merge existing online user's profile is a clear benefit for both system and user. For example, a mashup might gather and align user data from different social networking services in order to create a more comprehensive user profile during process runtime. It is a clear benefit to reuse such information for user adapted and user conditional activities since more is known about the user and less effort is required. In later sections we will explain how the platform supports flexible mashups not only in the context of user profiling.

3.3 Templates

In COOPER, a further support to process definition is given by the availability of process templates, i.e., process models that include the possibility of tagging some of the activities as mandatory. Mandatory activities must be executed in any legal enactment of the process (hence, for example, they cannot be directly or recursively included within disjunctive steps). When a template is copied into a process, the process inherits these constraints; therefore, mandatory template activities cannot be removed from such processes, neither due to their deletions, nor due to the creation of steps such that the mandatory activity can be bypassed. Moreover, if mandatory activities are related by a precedence relationship, then that precedence relationship must be preserved when the process is modified [11]. Setting an activity as mandatory is a choice of the template designer, who can constrain the way in which all processes derived from the template may evolve.

3.4 Architecture

Figure 1 illustrates how the features described in the previous section are composed into one comprehensive Web platform where different modules interoperate and make use of data and metadata. Process definition is performed via a process editor, a Web frontend that makes use of a predefined activity type library and, possibly, of existing process/template models. Process execution is performed via the COOPER's collaboration environment, which leverages on the hypertext front-ends of the predefined activity types to allow users to produce and consume process data in form of resources stored in the resource repository. Process advancement is then governed by the stored process definitions, which are interpreted during process execution by a dedicated process engine that contains the necessary application logic to maintain the running processes' metadata and, hence, to drive the activity flow in the collaboration environment. During the execution of a process, it is possible to check the status of the process and of the single activities composing the process by means of the process monitor. Figure 1 also highlights the competences of the individual actors in the COOPER platform. The activity designer identifies and designs the activity types that are available in the platform. The process designer then instantiate the activities types and defines process templates. The process designer can also be a team leader that wants to organize the work of his/her team. The users then perform the actual work or learning tasks. Users can also play the role of process designers. They are indeed enabled to define new processes by extending templates, or by composing new models from scratch. They are in any case allowed to modify template based process definitions during runtime, after the process has been launched, with the only limit of not violating the template constraints that the process may hold by definition.

3.5 Mashups

Despite the advantages that activity types introduce for the definition of flexible processes, the development of activities, as proposed by the COOPER approach, is still a time-consuming aspect, which requires Web developers to produce and integrate new Web pages required for the accomplishment of the activity. Activity types are strongly integrated within the application logics (they are pieces of the application front-end). Therefore, the introduction of new activity types requires extending the application code and releasing and deploying a new version of the platform. Since it is evident the difficulty of providing libraries of atomic ac-

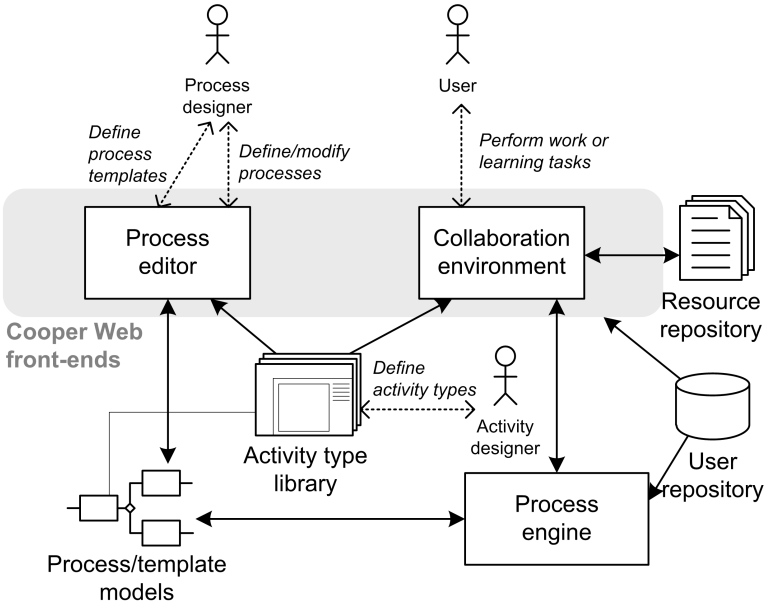

Figure 1: COOPER framework's architecture, supporting the definition and execution of flexible processes.

tivities able to cover all the possible and future needs of end-users, it is necessary to provide easy mechanisms to extend the library through external services and Web pages. Given the current emerging trend in Web applications development, we here explore the use of mashups to facilitate the definition and integration of new user activities in the process. We also let process designers and end-users to easily customize such activities during process execution.

A mashup is a Web application that combines services coming from heterogeneous sources [15]. When combined in a mashup, services not initially conceived to coexist with others allow users to have insights and make decisions based on the resulting combined data, generally conveyed through a single Web page. In our approach a mashup can be thought of as equivalent to a process node, i.e., an activity in a collaborative process. A process can thus incorporate one or more mashups. The next sections describe the mechanisms that we have defined for integrating process design and process engine on one side, and mashup models and tools on the other side.

3.6 Container Activities

To achieve the goal of integrating external mashups with process tools, we extend the Activity Type library with a new type: the Container Activity. It is a 'generics', based on two main concepts: (1) it exposes some properties that are required for its integration within a process (e.g., start and end time, enrolled process actors, etc.), and therefore can be used for process definition; (2) it is a generic container that can encapsulate user-defined mashups and, more in general, any user-defined Web page.

While defining a process, one or more CAs can be used when the ready-to-use activity types do not match the requirements of some user activities. The process designer can define a mashup (see next section) and encapsulate it within the container, by simply specifying the local or external URI of the application. Furthermore, s/he can characterize the output modality during the configuration of the activity, i.e. for external URI the explicitly output resource declaration to be produced, while for local URI the automatically definition of a resource as activity output. Such automation for local URI is reached using the abstract model of Mixup (see next section); within the abstract model it is possible to define which parameters of the components will be considered output of the activity and then automatically insert them into a textual resource as activity output. In details, a specific tag $< output >$ will be inserted in the abstract model any time the parameter of an event (see next section) of interest will be part of the output of the activity. At run time the Integration Engine will recognize such tag and will put into a textual resource the parameter values. This resource will be input of the next activities.

The encapsulation of the actual application supporting the activity execution can even be delayed at run-time: the process designer just configures the container properties that are necessary for process enactment, e.g., the enrolled user, the resources to be produced as activity output, and so on. The actual mashup for the activity execution can be encapsulated during the process execution by the enrolled actor.

3.7 Mashup Definition

When a process designer needs to instantiate a container activity with an external mashup s/he may use two different

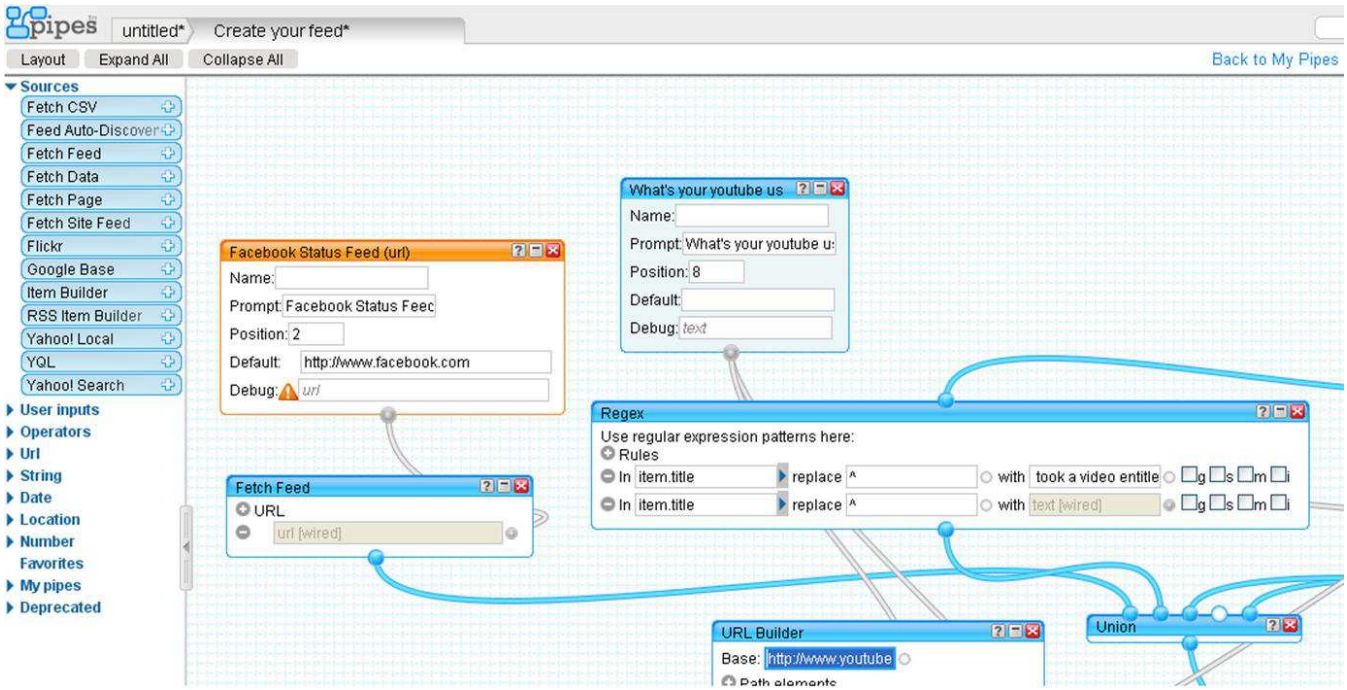

Figure 2: Yahoo!Pipes workbench interface with a User Pipe that gathers user profile data from multiple online social networks.

solutions: i) to create a mashup from scratch or ii) to reuse a mashup gathered from an external composition (or any other ready-to-use Web application).

In the first case, we assume the adoption of Mixup [15, 24], a mashup tool that supports integration at the presentation level, that is, component's integration by combining their presentation front-ends, rather than their application logic or data. The goal is to build composite, integrated applications that leverage the components' individual UIs to produce composite applications, the mashups, possibly with rich and highly interactive user interfaces. Communication and synchronization among components mainly consists of event notifications by one or more components, which trigger operations into other components, causing a change in their state. In the Mixup editor, this is specified by simply drag&drop components in a canvas, and by specifying the UI integration logic visually drawing connections among components. Managing such communication is possible thanks to some descriptors for components and composition.

In Mixup each component is characterized by an abstract model that describes the components ingredients useful for the integration within the mashup, namely events, operations, and properties, which allow the component to expose its state and configuration parameters. Conceptually, a component is indeed characterized by a state, which defines what the composite application can see and control in terms of changes to the UI. The state can be complex and consist of multiple attributes (e.g., map location and zoom level). Events allow notification of state changes, while operations allow for querying and modifications of the state.

Within our integrated platform, once the mashup application is created, the Mixup execution engine stores it in a Mashup Library, where it can be then associated to a Container Activity within a process definition. The newly achieved activity type can be then reused for further process definitions; depending on the permissions set by the creator, it can also be shared with other users. The integration of a mashup gathered from an external composition tool (as for example Yahoo!Pipes) requires linking the CA with the external URI of the mashup.

Another feature enhancing flexibility is the possibility to modify the created Container Activities. Once the process is running and one Container Activity is ready for execution, the system gives the enrolled users the option to personalize the mashups associated with the activity. The initial mashup definition works as a template, which holds the indication about what can be changed. For each mashup built with Mixup, it is indeed possible to define which elements are optional and which are mandatory, by means of suitable tags introduced into the components' descriptors. This gives some guarantees about the validity of the process, even when the end-user introduces some changes.

3.8 User Profile Mashups

By supporting the free design of activities, COOPER enables deduction of user profiles also by mashing up different

data streams in RDF or RSS-format by for example utilizing Yahoo!Pipes. Processing the RSS data by utilizing Yahoo!Pipes enables the usage of a huge amount of structured user data on the web. Different RSS streams are syndicated to so-called User Pipes [2].

A container activity mashup can simply be the output of some user pipe. To help creating these pipes, the Yahoo Pipe editor, provides an easy drag&drop user interface to process, combine, and perform various operations on data streams. It allows the common non expert semantic user to visually create profile reasoners. The benefit of this approach is that these user profile streams can be reused to build other reasoners. Yahoo Pipe editor permits the sharing of the pipes with the community. Potential 'pipes creators' are able to reuse and extend existing published pipes which, in our focus, allows for flexible and extensible user pipes.

Figure 2 shows the Yahoo Pipes editor interface and a user pipe that combines information from different Social Media services. In this example the user pipe retrieves and combines in a single RSS streams selected from Facebook user status update, blog feeds, news searches, Twitter posts, Last.fm user updates, Flickr updated photos and Youtube updated videos. Yahoo Pipe editor not only provides the interoperability of these data providers but also a variety of logical operations to process the information and build the desired structured output.

By combining mashups and thus enabling the use of the so called user pipes, COOPER reaches a new level in the field of Workflow Management Systems, providing not only interoperability with Web resources as mashable activities but also the potential benefit of *user adapted activity* and *user conditional activity*.

4. ENGINE INTEGRATION

At run time, the integration between the process execution engine and the mashup engine is achieved thanks to an intermediate module, in charge of activating the execution of mashups (both internally and externally defined) when a user activity corresponding to a CA must be executed. As shown in Figure 3, when a CA occurs as Select Hotel, the Builder module (a servlet) queries the internal repository looking for the mashup associated with the current CA. Then the Builder servlet composes the standard CA layout with the HTML/Javascript content of the mashup. Finally, the so-instantiated CA is presented to the user for execution. During the execution of the activity:

- The Mixup middleware manages the execution of the mashup. It ensures component communication by means of an event broker supervising a set of event listeners that map state change events, generated by one component, onto operations (i.e. state change requests) of other components.
- The Builder module then manages the process variables and resources that are manipulated by the current CA and that need to be preserved for successive activities. For example, it provides support to the application client for all the operations with the server filesystem, such as saving and updating a resource (e.g., file), and saving the status activity. In addition, it is in charge to store all the parameters of an event mashup tagged with $< output >$ in a textual resource as output activity.
- The Modifier module enables the change of optional components, preventing that one for the mandatory components. Mixup will let the user add new components, which will be stored with policy optional. In case of changes, Modifier will be in charge to update the XML files.

5. USE CASE SCENARIO

To better understand our approach, we will consider two simple scenarios that require the collaboration of multiple actors. We will describe firstly with a non-adaptive approach and later we will demonstrate how the user profile based activities can be beneficial for the process, tuning the scenario into an adaptive workflow process with adaptive activities.

5.1 Scenario I-A

In the first scenario imagine a team leader wants to suggest to team components some events e.g., conferences to attend. Team components select one conference and search for a hotel in the event location. Later, they ask the organization treasurer for the monetary availability. As represented in Figure 3, the process is composed by three activities: (1) an activity is related to the team leader who has to suggest a listing of conferences to attend; (2) a CA associated to the employee, who must select the conference and book a hotel; (3) an activity associated to the Treasurer, who has to check the cash and allocate, if available, the amount request. The treasurer will conclude the activity with a confirmation or negation message sent to the employee.

Suppose that the definition of this process requires the inclusion of a CA for the hotel booking activity and that the corresponding activity can be supported by four components that we suppose are already registered in the platform: the service "Find a Conference" supplying the list of the conferences indexed by DBWorld, the service "Hotel Retrieve" supplying a list of hotels for a given place of interest, the Flickr.NET component to display the images for a given hotel, and Google Maps to show the directions to a given address or point of interest. When the user selects a conference from the conference listing component, the hotel listing will be updated and will display all the hotels in that area, the image displayer will show an image of the selected hotel while the map will display its location.

5.2 Scenario I-B

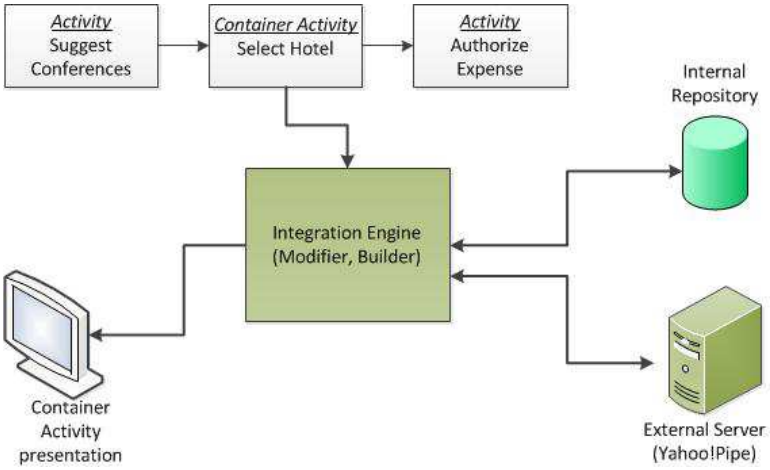

Figure 3: Integration framework.

For the adaptive version of the scenario imagine we have a Container Activity that consists in a User Pipe that gather user information provided by LinkedIn (a business-oriented professional social network popular among researchers). With the information from each team member, the first activity 'suggest a listing of conferences to attend' could be automatically performed by the system as the CA for the conference list would be adapted to each user's preference (user adapted activity). The team leader has no long the burden of suggesting upcoming conferences for each one of the team members. The same is applied to the hotel selection once the listed hotels are according to the conference date and location. In addition, the users' history of previous hotels' chains choices and other users' preferences can be applied to reduce the recall of options while increasing the precision.

5.2.1 Scenario II-A

In a second scenario imagine a team working in a business project that requires the involvement of professionals from different fields. A first version project proposal must be approved by the team leader at certain point of time. After that, there must be a i) proper description of the necessary technical work, ii) a description of the material and people involved, iii) a description of costs and iv) a description of the marketing planning. After these activities, a new activity is assigned to the team leader to review all the documentation to continue the project development.

Suppose that the definition of the CA is a mashup utilizing Google Docs API , allowing document editing and versioning control. The same CA can be reused for the four activities previously listed if the necessary actions are the same; however each activity refers to a very specific group of team members. So it is up to the team leader to approve the proposal in the first instance and later assign the following activities to team members - notice that this is a split activity in the workflow that turns the process in 4 different parallel sub-processes. Continuing in each activity, developers would document the technical issues, technicians and personal would describe the necessary material and people, treasurer would calculate the finances and the marketing members would describe market strategies. Once each team has finished each respective assignment, the workflow joints back together to an approval activity for the team leader.

5.2.2 Scenario II-B

In this scenario, the four activities that come after the same single split activity are not requirements to each other, neither they are executed by the same individuals. The split activity is not an ordinary conditional activity - conditional activities depend on the output of an activity to choose the next step. Here it is a split activity that doesn't exploit any computational power from the system. After the end of this activity the team leader must assign team members (one by one) to each of the following activities.

Exploiting the CA approach, it is possible to create alongside of the project proposal approval mashup, one User Pipe mashup that accesses the company's team member's profile. This can be merged with online profiles from each user for a better profile reasoning. Now the team leader has the necessary information and the ability to define conditional activities that redirect the team members based on their profiles. The burden of assigning each team member (or groups of team members) to each next activity in the workflow becomes an automatic system matching decision according to the rules defined by the leader.

For the next activities, each team member, when logged, is automatically assigned to the appropriate activity (one or more) according to his/her profile. In the workflow outlook there is not a simple split activity anymore. The mashed up activity combines user identity and the system's computational capability in profile reasoning to create the previously described *user conditional activity*.

6. CONCLUSIONS

In this paper, we have presented the basic ideas behind COOPER: a reference model for flexible processes definition and execution, together with its platform which allows, through the use of mashup technologies, to create user personalized process activities. We have extended the idea of the mashups activities to exploit the power of the available Web 2.0 data (and combine it) to enhance and provide adaptability on the workflow level. Through mashups activities we demonstrated how it is possible to combine and make use of online user profiles (user pipes) in the process.

These possibilities lead us to extend COOPER activities library with user profile based activities. Once it is feasible the reasoning of existing online user profiles we can reuse and apply this knowledge at the workflow level to i) guide the subject user through a split into different paths in the workflow (user conditional activity) and ii) prescribe the user activities pre-configured to the user's preferences or needs (user adapted activity).

COOPER is an ongoing work which is advancing to the new flexibility provided by the web services and mashups. Our first evaluations covering the creation of process and the use of mashups activities has shown us significant results on productivity impact, however the collected data and interviews were before the introduction of user profile based activities, thus out of the scope of this paper.

In our current stage we are working on improving the usability of user interfaces at design time and the evaluation of the proposed user profile based activities. Moreover we aim to provide basic user pipes for common process manages scenarios and have it public accessible through Yahoo!Pipes. From our future results we expect to find additional requirements and success factors for building a new class of Adaptive Workflow Management System that aggregates the well known Workflow Management System and Adaptive Systems.

Acknowledgment

This research has been co-funded by the European Commission within the eContentplus targeted project OpenScout, grant ECP 2008 EDU 428016 (cf. http://www.openscout.net).

7. REFERENCES

[1] *Blackboard. âĂIJBlackboard Academic Suite"*. 2007.

[2] F. Abel, D. Heckmann, E. Herder, J. Hidders, G.-J. Houben, D. Krause, E. Leonardi, and K. van der Slujis. A framework for flexible user profile mashups. In *Proceedings of International Workshop on Adaptation and Personalization for Web 2.0 (AP-WEB 2.0 2009)*, CEUR Workshop Proceedings, pages 1–10, Trento, Italy, June 22 2009. CEUR, Tilburg, Aachen.

[3] M. Adams, D. Edmond, and A. H. M. T. Hofstede. The application of activity theory to dynamic workflow adaptation issues. In *In Proceedings of the 2003 Pacific Asia Conference on Information Systems (PACIS 2003*, pages 1836–1852, 2003.

[4] A. Ankolekar, M. KrÃűtzsch, T. Tran, and D. Vrandecic. The two cultures: Mashing up web 2.0 and the semantic web. In *PROCEEDINGS OF THE 16TH INTERNATIONAL CONFERENCE ON WORLD WIDE WEB. 2007 MAY 7-8.* ACM Press, 2007.

[5] Aroyo, Dolog, Houben, Kravcik, Naeve, Nilsson, and Wild. Interoperability in pesonalized adaptive learning. In *J. Educational Technology and Society 9 (2)*, 2006.

[6] Assad, Carmichael, Kay, and Kummerfeld. Personisad: Distributed, active, scrutable model framework for context-aware services. pages 55–72, 2007.

[7] Barthelmess and Ellis. The neem platform: An evolvable framework for perceptual collaborative applications. In *J. Intell. Inf. Syst. 25, 2*, pages 207–240, 2005.

[8] Bongio, V. Bruggen, Ceri, Cristea, Dolog, Hoffmann, Matera, Mura, Taddeo, Zhou, and Zoni. Cooper: Towards a collaborative open environment of project-centred learning. In *Springer*, pages 561–566, 2006.

[9] Brickley and Miller. *FOAF Vocabulary Specification 0.91. Namespace document, FOAF Project.* 2007.

[10] Carell, Herrmann, Kienle, and Menold. Improving the coordination of collaborative learning with process models. In *CSCL 2005. Computer Supported Collaborative Learning*, pages 18–27, 2005.

[11] Ceri, Daniel, Matera, and Raffio. Providing flexible process support to project-centered learning. In *IEEE Trans. on Knowl. and Data Eng. 21, 6*, pages 894–909, 2009.

[12] Cherbakov, Bravery, Goodman, and Baggett. Changing the corporate it development model: Tapping the power of grassroots computing. In *IBM Systems Journal 46, 4*, 2007.

[13] Cuzzocrea, A., De Coi, J.L., Fisichella, M., and Matera, M. (2011). Graph-based matching of composite OWL-S services. In *DASFAA Workshops*, pages 28–39.

[14] Cuzzocrea, A. and Fisichella, M. (2011). Discovering semantic Web services via advanced graph-based matching. In *IEEE SMC*, pages 608–615.

[15] Daniel, Yu, Benatallah, Casati, Matera, and Saint-Paul. Understanding ui integration: A survey of problems, technologies, and opportunities. In *IEEE Internet Computing 11 (3)*, pages 59–66, 2007.

[16] Fisichella, M. and Matera, M. (2011). Process flexibility through customizable activities: A mashup-based approach. In *ICDE Workshops*, pages 226–231.

[17] Frias-Martinez, Magoulas, Chen, and Macredie. Modeling human behavior in user-adaptive systems:

Recent advances using soft computing techniques. In *Expert Systems with Applications 29*, pages 320–229, 2005.

[18] Greiner, MÃijller, Rahm, Ramsch, Heller, and LÃűffler. Adaptflow: Protocol-based medical treatment using adaptive workflows. In *Methods of Information in Medicine*, pages 80–88, 2005.

[19] Heckmann, Schwartz, Brandherm, Schmitz, and V. Wilamowitz-Moellendorff. Gumo - the general user model ontology. In *The 10th Int. Conf. on User Modeling*, pages 428–432, 2005.

[20] MÃijller, Greiner, and Rahmr. Agentwork: A workflow-system supporting rule-based workflow adaptation. In *Data and Knowledge Engineering*. Elsevier, 2004.

[21] Nodenot, Marquesuzaa, Laforcade, and Sallaberry. Model based engineering of learning situations for adaptive web based educational systems. In *WWW 2004 .The 13th international World Wide Web Conference on Alternate Track Papers and Posters*, pages 94–103, 2004.

[22] Paramythis and Reisinger. Adaptive learning environments and e-learning standards. pages 181–194, 2004.

[23] Stewart, Celik, Cristea, and Ashman. Interoperability between each user models. In *APS 2006*, 2006.

[24] Yu, Benatallah, Saint-Paul, Casati, Daniel, and Matera. A framework for rapid integration of presentation components. In *WWW 2007, ACM Press*, pages 923–932, 2007.

[25] Yudelson, Brusilovsky, and Zadorozhny. A user modeling server for contemporary adaptive hypermedia: an evaluation of the push approach to evidence propagation. In *The 11th International Conference on User Modeling*, pages 27–36, 2007.